STOCHASTIC ANALYSIS OF

MULTIPLE-PASSBAND SPECTRAL CLASSIFICATIONS

SYSTEMS AFFECTED BY OBSERVATION ERRORS

(NASA-CR-164366) STOCHASTIC ANALYSIS OF MULTIPLE-PASSBAND SPECTRAL CLASSIFICATIONS SYSTEMS AFFECTED BY OBSERVATION ERRORS (University of South Florida) 31 p HC A03/MF A01 CSCL 05B G3/43

N81-24505

Unclas 29496

Prepared

for

Flight Electronics Division/NASA

Langley Research Center

Hampton, Virginia 23665

Under NASA Research Grant NAG1-9(CLC)

Chris P. Tsokos, Ph.D

University of South Florida

December 1980

STOCHASTIC ANALYSIS OF MULTIPLE-PASSBAND SPECTRAL CLASSIFICATIONS SYSTEMS AFFECTED BY OBSERVATION ERRORS

TABLE OF CONTENTS

STOCHASTIC ANALYSIS OF MULTIPLE-PASSBAND SPECTRAL CLASSIFICATIONS SYSTEMS AFFECTED BY OBSERVATION ERRORS

ABSTRACT

The problem of classifying targets viewed by a "push-broom"-type multiple-band spectral scanner by means of algorithms suitable for implementation in high-speed online digital circuits is considered. A class of algorithms suitable for use with a pipelined classifier is investigated through simulations based on observed data from agricultural targets. The time distribution of target types is shown to be an important determining factor in classification efficiency.

LIST OF TABLES AND GRAPHS

STOCHASTIC ANALYSIS OF MULTIPLE-PASSBAND SPECTRAL CLASSIFICATIONS SYSTEMS AFFECTED BY OBSERVATION ERRORS

I. INTRODUCTION

In considering the problem of rapid, efficient classification of images by an online processor in an earth-resources satellite, the complexity and speed of the processor represent extremely important constraints on the classification algorithms which can be used. In the present study, we consider classification algorithms which may be used with digital devices organized into a pipeline processor designed to operate synchronously with a "pushbroom"-type spectral scanner. This particular study is based on analyses and simulations of a three-stage classifier designed to discriminate among the crops represented in the data contained in LARS tape, that was obtained from Purdue University, but the methodology is easily applied for any number of stages.

The architecture of the system is described in schematic form in Figure 1. At time t, the spectral scanner outputs the output from the target which is scanned by filter 1 at time t, the output from filter 2 at time t (which represents the signal from the same target which filter 1 scanned at time t-1); and the output from filter 3 at time t (corresponding to the target scanned by filter 1 at time t-2). These signals are represented by $x_1(t)$, $x_2(t-1)$, and $x_3(t-2)$.

The indices follow the convention, which we adopt throughout this investigation, that t represents the time at which the target being analyzed crossed filter 1. At each stage, a processor uses the signal from the corresponding filter and information stored in a pattern library in system memory to produce a vector $\underline{C}_i$, $i = 1, 2, 3$, of information for classification. In the systems which we consider this information will always be a Bayesian estimator of the vector of a posteriori probabilities for each type of source. The three algorithms considered in the present study can be implemented by processor which can complete the updating of $\underline{C}$ in a time on the order of a few hundred microseconds, thus permitting synchronous operation.

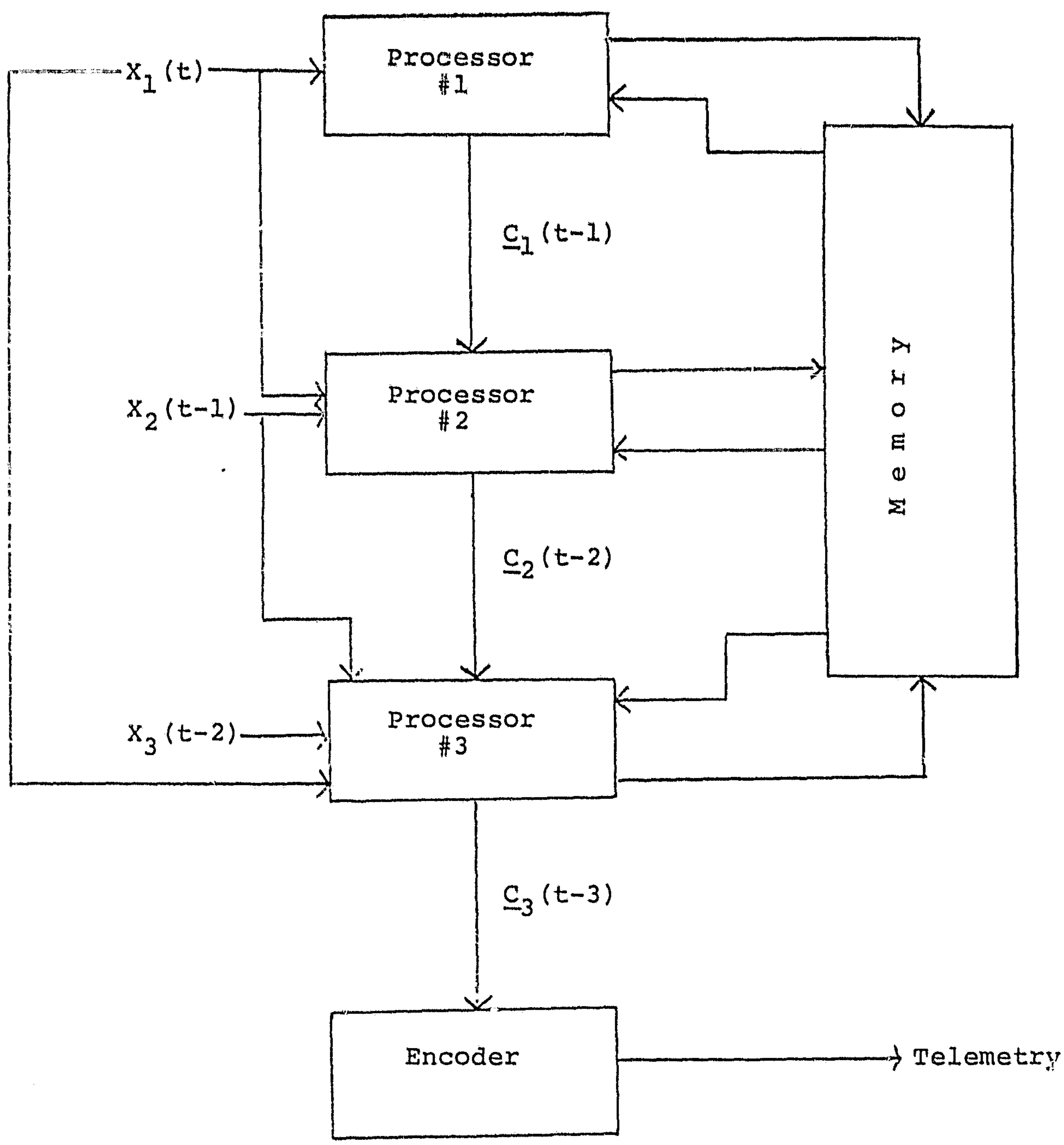
In Section II we discuss the actual crop data that was used in the present study. A description of the stochastic model for which the classification algorithms are based on is given in Section III. In Section IV, we give a detail description of the Bayesian classification algorithms that includes the results of the two different type of classifiers that were employed in the present study. Summary and recommendation of the present investigation are presented in Section V.

A listing of the software that were developed for the present investigation is given in Appendix A of this report.

FIGURE 1

Schematic Diagram of 3-Stage Pipeline

Processing Classifier



II. CALIBRATION DATA

The actual data on which the statistics in this report are based are the spectral scans of crops contained in LARS tape, that was obtained from Purdue University. Of the spectra on this tape, 2434 are optical spectra of crops for which 60 or more observations are available. We considered 12 possible filters, each of which has a rectangular passband spanning six consecutive wavelengths on the tape, and selected those three filters which provided the highest entropy for the joint distribution of x_1, x_2, and x_3, thus maximizing the total information reaching the classifier. Since the correlation between adjacent wavelengths is very high, it is not surprising that the filters chosen were widely spaced. The three filters chosen, in order by decreasing conditional entropy of the output distribution, were:

Filter 1:	Wavelengths 4 - 9
Filter 2:	Wavelengths 24 - 29
Filter 3:	Wavelengths 59 - 64

The crops considered, and the number of observations for each, are given in Table 1, which also contains χ^2 (chi-squared) statistics for the test of normality described below.

In order to determine whether linear discriminant analysis should be considered as a classification algorithm,

we used a statistical test of multivariate normality for the three-dimensional vectors of filter outputs. Let $\hat{\underline{\mu}}$ be the sample mean of $\underline{x}$, vector filter outputs, for a given source class, and let $\hat{S}$ be the sample covariance matrix. Then the random variable

$$\phi \equiv (\underline{x}-\underline{\mu})^T \hat{S}^{-1} (\underline{x}-\underline{\mu})$$

should have a χ^2 distribution with 3 degrees of freedom, and

$$P_{\chi^2}(\phi;3)$$

should be uniformly distributed. In order to test the uniformity of the distribution of this latter statistic, we divided the unit interval into ten equal subintervals and used the conventional χ^2 test (with 9 degrees of freedom) for the resulting one-way contingency table. It is seen from Table 1 that neither $\underline{x}$ nor log $(\underline{x})$ passed this test. We thus conclude that linear discriminant analysis is inappropriate and we must consider only nonparametric classification algorithms based on the empirical sample frequency tables.

TABLE 1

Tests for Goodness-of-Fit

of Normality

Crop	N	χ^2 for Normal	Lognormal
Alfalfa	82	3.1	5.8
Corn	297	27.3	15.8
Sorghum	290	15.7	28.3
Wheat	373	67.5	56.4
Unplanted & Fallow	1392	129.5	35.5

Note: The chi-squared test fails at the .01 level of significance for all crop except alfalfa.

III. STOCHASTIC MODEL

The classification algorithms which we consider in the present investigation are based on a Markovian process type of schemes in which the probability of source s(t) at time t is governed by the recurrence relation

$$\underline{p}\,(t) = (1-\alpha)\,\underline{p}(t-1) + \alpha\underline{\pi}.$$

Here $\underline{p}$ is the vector of probabilities for each source type, $\alpha\varepsilon[0,1]$ is the one-step transition probability, and $\underline{\pi}$ is the vector of unconditional probabilities. In the absence of information concerning the relative abundances of the various crops we assume that

$$\pi^T = (\tfrac{1}{5}\ \tfrac{1}{5}\ \tfrac{1}{5}\ \tfrac{1}{5}\ \tfrac{1}{5})$$

for the five sources used in these simulations. We test each of the classification algorithms on three 10,000-point simulations. The valves used in generating the pseudorandom samples are $\alpha = .2$, $\alpha = .5$, and $\alpha = .8$, representing high, intermediate, and low persistence, respectively.

The information used in classification is initially decoded by comparison with threshold values as follows:

$$\phi_1(t) = i \quad \text{iff} \quad \tau_{i1} \le x_1 < \theta_{i1}$$

$$\phi_2(t) = j \quad \text{iff} \quad \tau_{j2}\,(\phi_1) \le x_2 < \theta_{j2}(\phi_1)$$

$$\phi_3(t) = k \quad \text{iff} \quad \tau_{k3}\,(\phi_1,\phi_2) \le x_3 \le \theta_{k3}(\phi_1,\phi_2).$$

defined for

$$1 \le i \le 5,\ 1 \le j \le 5,\ 1 \le k \le 5.$$

Note that

$$\tau_{11} = \tau_{12} = \tau_{13} = -\infty$$

and

$$\theta_{11} = \theta_{12} = \theta_{13} = \infty.$$

The thresholds are chosen such that $\underline{\phi}$ is uniquely defined and the entropy of the distribution of $\underline{\phi}$ is maximized. Note that the thresholds used for discretization at each step depend on the results of the previous step. This procedure considerably increases the information contained in $\underline{\phi}$, since the three filter outputs $\underline{x}$ are not stochastically independent.

Furthermore, we assume that the probability distribution of $\underline{\phi}$ is defined by

$$\underline{\phi}(t) = \sum_{i=1}^{5} p_i(t)\ f_i\ (\underline{\phi}).$$

This assumption amounts to ignoring serial autocorrelation of the $\underline{\phi}$'s whenever $s(t) \neq s(t-1)$. We do not ignore the dependence between $\underline{\phi}(t)$ and $\underline{\phi}(t-1)$ due to the Markovian dependence of $\underline{p}(t)$ on $\underline{p}(t-1)$. This assumption may understate the extent of serial autocorrelation in the signal,

that is, overestimate the source entropy, but it seems the most reasonable assumption to use in the absence of sufficiently detailed observational data. The relevant probabilities could be estimated from observational runs in which groups of consecutive observations have not been averaged during preliminary processing.

The actual generation of sample points for simulations is performed according to the following algorithm based on the stochastic process described above:

1. Choose $s(1)$ at random from distribution π.

2. For t from 2 to 10,000 perform steps 3 through 5.

3. Generate $u \epsilon (0,1)$ with uniform distribution.

4. If $u \leq \alpha$, choose $s(t)$ at random from distribution π; otherwise set $s(t) = s(t-1)$.

5. Choose $x(t)$ at random from the actual observations for source type $s(t)$ in the data tape.

The pseudorandom runs of 10,000 points generated for each of the three values of α by this algorithm are used in all of the tests of classification algorithms described below.

IV. CLASSIFICATION ALGORITHMS

As was pointed out in the discussion of the observational data, the filter outputs $\underline{x}$ fit both normal and lognormal distributions so poorly that linear discriminant analysis is unsuitable for the problem at hand. For this reason, we have concentrated on Bayesian discrimination as a classification technique.

A Bayesian classification could be based on either nonparametric density estimates for each source class or on contingency tables from discretized data. The large amount of computations required for nonparametric density estimators argue against applying them for on-line image-analysis device. The high storage requirements and search times required for nearest-neighbor classification similarly appear to preclude the use of this technique in the present application.

Bayesian discrimination based on discretized scanner outputs requires only relatively modest amounts of memory and is well adapted for a pipeline architecture consistent with very rapid operation. A n-stage classifier for M categories based on a K-level discretization of each filter output requires only

$$Mk\left(\frac{k^{n-1}}{k-1}\right)$$

real values in memory. The basic mathematical operations

used are elementwise multiplication and addition of M-dimensional vectors, a circumstance which also favors implementation by parallel-processor systems.

We now define the Bayesian update operation as a vector-valued function of two vectors by

$$B_i(\underline{p},\underline{q}) = \frac{p_i q_i}{\sum_{i=1}^{M} p_i q_i} \quad .$$

Now let $f(i;j,k,l)$ denote the probability density of the event $\underline{\phi} = (j,k,l)$ given that the source is i. We define three sets of likelihood vectors by

$$\lambda_{1i}(j) = \frac{\sum_{k,l} \sum f(i;j,k,l)}{\pi(i)} \quad .$$

Note that,

$$\sum_j \sum_k \sum_l f(i;j,k,l) \equiv \pi(i)$$

with

$$\lambda_{2i}(j,k) = \frac{\sum_l f(i;j,k,l)}{\pi(i)\ \lambda_{1i}(j)}$$

and

$$\lambda_{3i}(j,k,l) = \frac{f(i;j,k,l)}{\pi(i)\lambda_{1i}(j)\lambda_{2i}(j,k)} \quad .$$

The vector of posterior probabilities for the categories based on a single observation $\phi = (j,k,l)$ could then be implemented as follows:

$$\underline{C}_1 = B(\underline{\pi}, \underline{\lambda}_1(j))$$

$$\underline{C}_2 = B(\underline{C}_1, \underline{\lambda}_2(j,k))$$

$$\underline{C}_3 = B(\underline{C}_2, \underline{\lambda}_3(j,k,l)).$$

This three-stage calculation is consistent with the proposed architecture of the classifier. The use of a three-step calculation offers no particular advantage for the single-observation classifier just described, but does enhance the speed of classifiers based on the more sophisticated algorithms described below.

To take advantage of the Markovian processes type of character of the assumed stochastic process, we may add a fourth computational stage

$$\underline{C}_4(t) = B(\underline{C}_3(t), (1-\alpha^*)\underline{C}_4(t-1) + \alpha^*\underline{\pi}),$$

where α^* is the estimated transition probability for the Markov process. Such a postprocessor increases the lag between observation and classification by one cycle time but does not affect synchronous operation if implemented in a pipelined system. The results of using classifiers based on $\alpha^* = 1, .8, .5$, and $.2$ on 10,000-point simulations with actual transition probabilities $\alpha = .8$, $.5$, and $.2$ are given in Table 2.

As one might expect, the classification is most efficient when $\alpha = \alpha^*$. The algorithm does, however, appear quite robust in that underestimating the transition probability does not severely degrade performance. We remark that $\alpha^* = 1$ corresponds to a memoryless classifier in which the fourth Bayesian update step is omitted. The use of the Markovian property improves on the results obtained with $\alpha^* = 1$ in all cases except $\alpha = .8$, $\alpha^* = .2$, for which the transition probability is grossly under-estimated. Even in this case, the degradation of performance is small.

A slightly more refined classification algorithm makes use of the fact that a small amount of forward information is always available. By time $t + 2$, when the image scanned by filter 1 at time t is ready to be classified, the data $\phi_1(t+1)$, $\phi_1(t+2)$, and $\phi_2(t+1)$ are also available. By increasing the memory requirement, we can use all of this information in classifying the source as follows:

Define

$$
\begin{aligned}
&\lambda^*(i;j_1,j_2,k_1) \\
&= \sum_{l} f(i,j_1,k_1,l) \Big[(1-\alpha) \sum_{k_2} \sum_{l_2} f(i;j_2,k_2,l) \\
&\qquad + \alpha \sum_{i_2} \sum_{k_2} \sum_{l_2} f(i_2;j_2;k_2,l_2) \Big] ,
\end{aligned}
$$

$$\lambda^{**}(i;j_1,j_2,j_3,k_1,k_2,l_1) =$$

$$\sum_{l} f(i;j_1,k_1,l_1) \Big[(1-\alpha)^2 \sum_{l_2} \sum_{k_3} \sum_{l_3} f(i;j_2,k_2,l_2) f(i,j_3,k_3,l_3)$$

$$+2\ \alpha(1-\alpha) \sum_{i_2} \sum_{l_2} \sum_{k_3} \sum_{l_3} f(i_2;j_2,k_2,l_2) f(i_2;j_3,k_3,l_3)$$

$$+\alpha^2 \sum_{l_2} \sum_{i_3} \sum_{l_2} \sum_{k_3} \sum_{l_3} f(i_2;j_2,k_2,l_2) f(i_3;j_3,k_3,l_3) \Big]$$

The classification algorithm can then be specified as follows:

$$\underline{C}_1(t) = B(\underline{\pi},\underline{\lambda}_1(\phi_1(t))),$$

$$\underline{C}_2(t) = B(\underline{C}_1(t),\underline{\lambda}^{**}(\phi_1(t-2),\phi_1(t-1),\phi_1(t),$$

$$\underline{C}_3(t) = \underline{B}(\underline{C}_2(t),\underline{\lambda}^{**}(\phi_1(t-2),\phi_1(t-1),\phi_{\underline{1}}(t),$$
$$\phi_2(t-1),\phi_2(t),\phi_3(t)),$$

and

$$\underline{C}_4(t) = \underline{B}(C_3(t),(1-\alpha)\underline{C}_4(t-1)+\alpha\underline{\pi}).$$

Results from this classifier are given in Table 3. While the improvement over the results of the simpler classifier evaluated in Table 2 are not large. The greater complexity may be justified by the small gain achieved in critical applications.

A graphical presentation of the probabilities of misclassification in the 10,000 point simulation test for type 1 and type 2 classifiers are given by Figure 2 and 3, for the actual transition probabilities, α, and the assumed ones, α^*.

TABLE 2

Number of Misclassifications

In

10,000 Point Simulation Test

Type 1 Classifier

α^* \ α	0.2	0.5	0.8
0.2	2687	4456	5470
0.5	3086	4129	4858
0.8	4070	4346	4664
1.0	4775	4732	4822

α^* = assumed transition probability

α = true transition probability

TABLE 3

Number of Misclassifications

In

10,000 Point Simulation Test

Type 2 Classifier

α^* \ α	0.2	0.5	0.8
0.2	2420	4356	5427
0.5	2782	3962	4832
0.8	3757	4181	4630
1.0	4775	4732	4822

α^* = assumed transition probability

α = true transition probability

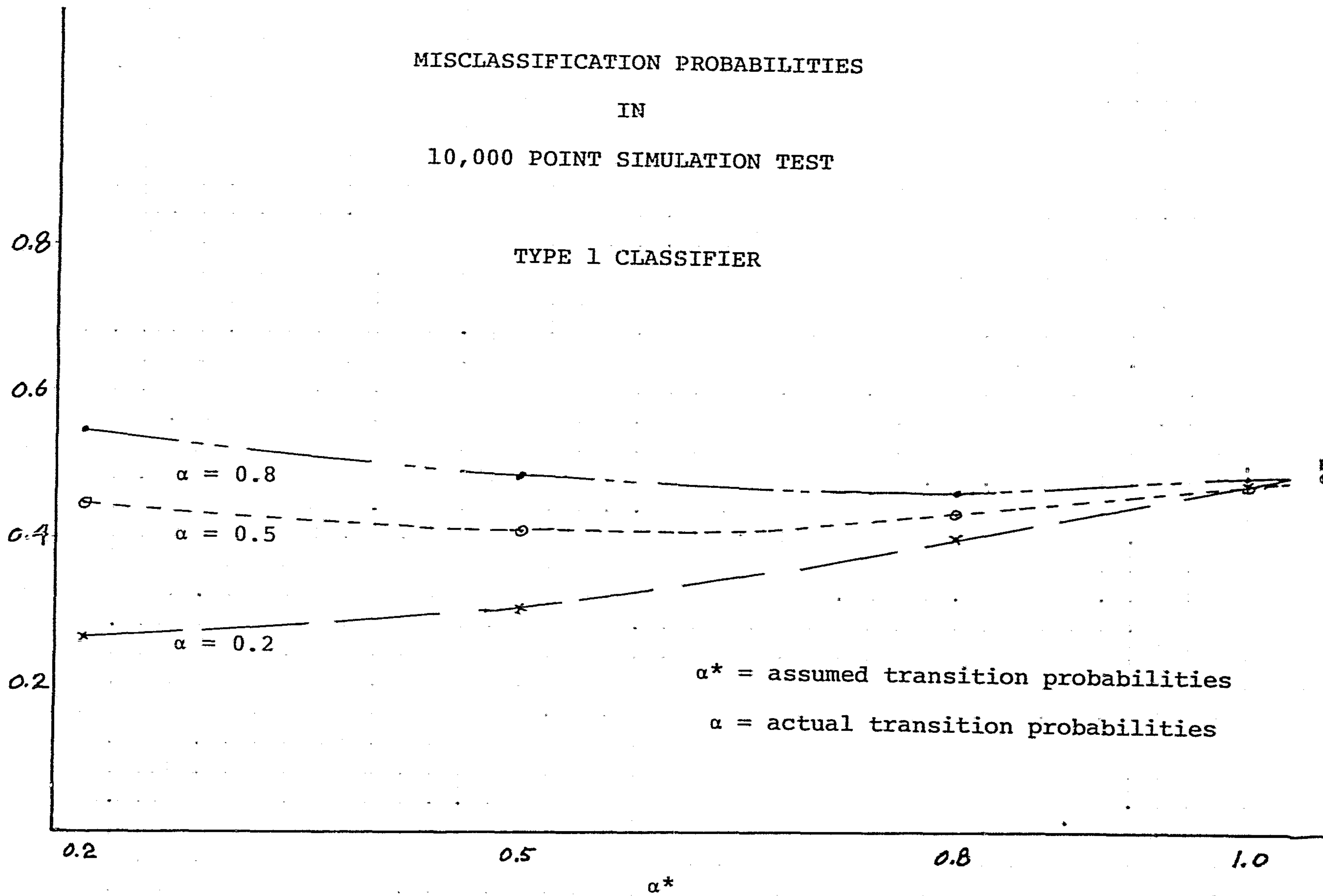

MISCLASSIFICATION PROBABILITIES
IN
10,000 POINT SIMULATION TEST
TYPE 1 CLASSIFIER
0.8
0.6
0.4
0.2
α = 0.8
α = 0.5
α = 0.2
α* = assumed transition probabilities
α = actual transition probabilities
0.2
0.5
0.8
1.0
α*

FIGURE 2

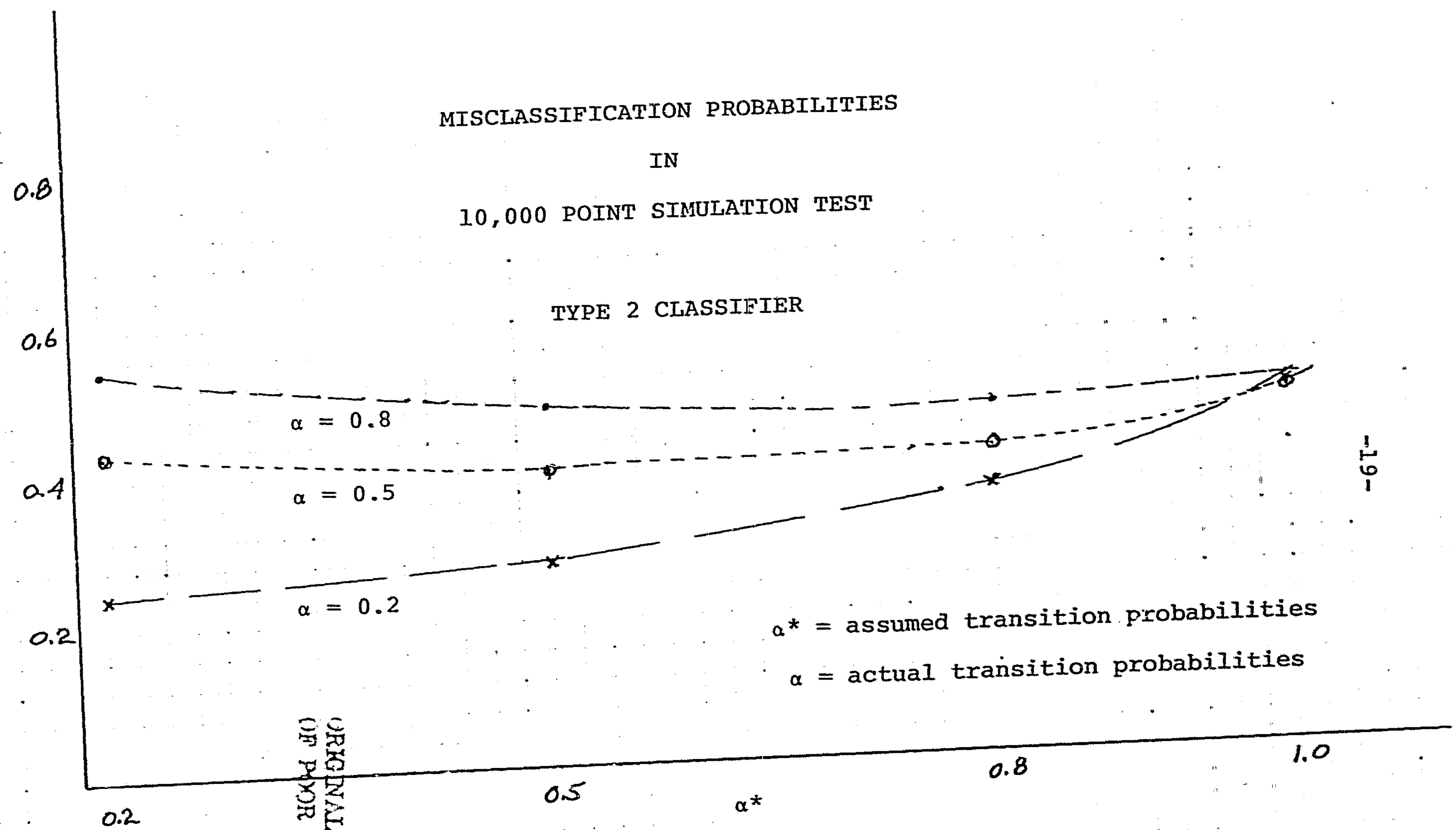

MISCLASSIFICATION PROBABILITIES
IN
10,000 POINT SIMULATION TEST
TYPE 2 CLASSIFIER
0.8
0.6
0.4
0.2
α = 0.8
α = 0.5
α = 0.2
α* = assumed transition probabilities
α = actual transition probabilities
0.2
0.5
0.8
1.0
α*

FIGURE 3

V. SUMMARY

The results presented, based on a classifier and a Markovian target-type transition process, show that the reduction of source entropy due to a tendency for adjacent targets to be of similar type can be effectively exploited as a source of information to improve the efficiency of a multistage image classifier. It is to be expected that an effect this large will generalize to other Bayesian classification algorithms and other transition processes.

The most serious limitation on the efficiency of the classifiers arises from the necessity of using relatively coarse contingency tables to estimate the posterior probabilities of the source types. This defect could be overcome either by using a larger corpus of observations to refine the empirical frequencies or by developing an analytical model for the conditional distribution of the filter outputs from each type of target. The very poor goodness-of-fit results given in Table 1 indicate that this will probably be a difficult task.

VI. RECOMMENDATIONS

As a consequence of the demonstrated potential for exploiting temporal coherence of the sequence of target types scanned in order to reduce classification error, the investigation of the underlying stochastic process should be considered as a research objective. Studies of the spatial coherence properties of the mix of target types on a scale from several hundred meters to several kilometers would be useful for this purpose.

The characterization of the probability density function of the spectrum of each source type is also a necessity for achieving efficient discrimination in practical applications. While the standard multivariate normal and lognormal densities provide a poor fit, it is likely that some effective approximation in terms of a superposition of simple density functions can be achieved when enough data on a given source type becomes available. The application of cluster analysis to large data samples would be useful in this connection.

APPENDIX A

Software for

STOCHASTIC ANALYSIS OF MULTIPLE PASSBAND SPECTRAL CLASSIFICATIONS SYSTEMS AFFECTED BY OBSERVATION ERRORS

```
//ZUUSIGC JOB (,,299,10),'  ROGI  ',
// MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=Z00513
/*SETUP DISK=HV012,TAPE=1
/*JOBPARM I=1,ROOM=K028
//ONLY EXEC FORTGCLG,PARM.FORT='MAP,ID',PARM.LKED='MAP,XREF',
//   LIBS='USF.CO.P14972.X12.LOGLIB'
//FORT.SYSIN DD *
      IMPLICIT INTEGER*4(A-Y)
      IMPLICIT LOGICAL*1(Z)
      COMMON/ISAR/H(3000),M(3000),N(3000),LN,LH,LR,T,ZE,ZX
      CALL RE
      CALL KE
  100 CALL K
      GOTO(110,120,130,140,150),T
  110 CALL FBICM(H,N,LN)
         LH=LN
      CALL RE
      CALL K
      GOTO 200
  120 CONTINUE
  130 CONTINUE
  150 CALL ER
      CALL RE
      GOTO 100
  140 GOTO 999
  200 GOTO (210,220,230,240,250),T
  210 CALL ER
      GOTO 100
  220    CALL FBICM(R,N,LN)
         LR=LN
      CALL RE
      CALL K
      GOTO 300
  230 CONTINUE
  250 CONTINUE
      CALL ER
      CALL RE
      GOTO 100
  240 CALL ER
      GOTO 999
  300 GOTO(310,320,330,340,350),T
  310 CALL HR
      GOTO 100
  320 CONTINUE
  350 CALL ER
      CALL RE
      GOTO 100
  330 CALL HR
      CALL RE
      GOTO 100
  340 CALL HR
      GOTO 999
  999 CALL GETIR(T,J,L,ZE,ZX)
      STOP
      END
      SUBROUTINE K
      IMPLICIT INTEGER*4(A-Y)
      IMPLICIT LOGICAL*1(Z)
      DIMENSION KL(3)
      COMMON/ISAR/H(3000),M(3000),N(3000),LN,LH,LR,T,ZE,ZX
      DATA KL/1200,1248,1574/
      IF(.NOT.ZX)GO TO 5
      T=5
      GOTO 99
    5 IF(.NOT.ZE) GOTO 10
      T=4
      GOTO 99
   10 IF(N(1).LE.0)GOTO 15
      T=1
      GOTO 90
   15 IF(N(1).EQ.0) GOTO 20
      T=5
      GOTO 99
   20 T=5
      IF(N(2).EQ.1)T=2
      IF(N(2).EQ.2)T=3
      IF(T.EQ.5)GOTO 99
   90 IF(LN.NE.KL(T))T=5
   99 RETURN
      END
      SUBROUTINE ER
```

ORIGINAL PAGE IS OF POOR QUALITY

```
      IMPLICIT INTEGER*4(A-Y)
      IMPLICIT LOGICAL*1(Z)
      COMMON/IOAR/ N(8000),H(8000),R(8000),LN,LH,LR,T,ZE,ZX
 1    FORMAT('   ERROR   ',4I10,L2)
      WRITE(6,1)T,LN,N(1),N(2),ZE,ZX
      RETURN
      END
      SUBROUTINE RE
      IMPLICIT INTEGER*4(A-Y)
      IMPLICIT LOGICAL*1(Z)
      COMMON/IOAR/N(8000), H(8000),R(8000),LN,LH,LR,T,ZE,ZX
      XX=32000
      CALL GETRN(N,XX,LN,ZE,ZX)
      RETURN
      END
      SUBROUTINE WR
      IMPLICIT INTEGER*4(A-Y)
      IMPLICIT LOGICAL*1(Z)
      COMMON/IOAR/N(8000),H(8000),R(8000),LN,LH,LR,T,ZE,ZX
       DATA RC /0/
 1    FORMAT('   OUTPUT   ',7I6)
       RC=RC+1
      WRITE(6,1) RC,LH,H(1),H(2),LR,R(1),R(2)
 2          FORMAT(20A4)
            RN=LR/4
            HN=LH/4
            WRITE(6,2)(H(I),I=1,HN),(R(I),I=1,RN)
      RETURN
      END
//GO.XIT DD DSN=XXX,DISP=OLD,UNIT=TAPE,VOL=SER=LARS01,LABEL=(1,NL)
//GO.FT08F001 DD DSN=JSM.GG.P1472.$X12.LARS8,DISP=OLD
```

Data extraction from LARS tape using BSAM I/O.

```
**** TSO FOREGROUND HARDCOPY ****
DSNAME=Z00513.A.DATA                                    (LRP1     )

 DATA A;SET X.P2F;KEEP GCOL;
  DATA B;SET X.P2F;KEEP NS;
 DATA C;SET X.P2A;KEEP V1-V12;
PROC MATRIX;FETCH GG DATA=A;NG= NROW(GG);FETCH NS DATA=B;
DO K=1 TO NG;N=NS(K,);C=J(12,12,0);E=J(1,12,0); NR=N;
 LA:IF NR>0 THEN DO; NN=NR;IF NN>100 THEN NN=100;NR=NR-NN;
 FETCH X NN DATA=C;E=E+X(+,);C=C+X'*X;GOTO LA;END;
 E=E#/N;
 C=X'*X#/N-E'*E;C=C#N#/(N-1);EIGEN V R C; D=E//V'//R';
 OUTPUT D DATA=X.P2M;END;
PROC MATRIX;FETCH GG DATA=A;NG=NROW(GG);FETCH NS DATA=B;
DO K=1 TO NG;N=NS(K,);C=J(12,12,0);E=J(1,12,0); NR=N;
 LA:IF NR>0 THEN DO; NN=NR;IF NN>100 THEN NN=100;NR=NR-NN;
 FETCH X NN DATA=C;X=LOG(X);E=E+X(+,);C=C+X'*X;GOTO LA;END;
 E=E#/N;
 C=X'*X#/N-E'*E;C=C#N#/(N-1);EIGEN V R C; D=E//V'//R';
 OUTPUT D DATA=X.P2ML;END;
```

SAS Program for Calculation of Moments.

```
**** TSO FOREGROUND HARDCOPY ****
DSNAME=Z00513.A.DATA                                    (LRSP4    )

PROC SORT DATA=X.P2A OUT=A;BY V1 V5 ;
 PROC RANK DATA=A OUT=B GROUPS=4;VAR V1;
 PROC RANK DATA=B OUT=A GROUPS=4;BY V1 ; VAR V5;
 PROC SORT DATA=A OUT=B;BY V1 V5;
PROC RANK DATA=B OUT=A GROUPS=4;
   BY V1 V5;VAR V2-V4 V6-V12;
PROC SORT DATA=A OUT=X.P2R3;BY V1 V5 CODE;
DATA X.P213;SET X.P2R3;BY V1 V5 CODE;
ARRAY N(I) N1-N12; ARRAY F(J) F1-F48; ARRAY V(I) V1-V12;
RETAIN F1-F48 C; KEEP CODE GR N1-N12;
DO I=1 TO 12; J=12*V+I; F=F+1;END;
IF LAST.CODE THEN DO;
 DO GR=0 TO 3;DO I=1 TO 12;J=12*GR+I;N=F;END;OUTPUT;END;
 DO OVER F;F=0;END;
END;
DATA X.P2IN3;SET X.P213 END=EOF ;ARRAY N(I) N1-N12; ARRAY H(I) H1-H12;
 RETAIN H1-H12 0;
 DO I=1 TO 12;IF N>0 THEN DO;H=H+N*LOG(N);END;END;
 IF EOF THEN OUTPUT; KEEP H1-H12;
PROC PRINT DATA=X.P213 (OBS=8);
PROC PRINT DATA=X.P2IN3;
```

SAS Program to Calculate Source Entropy.

```
**** ISO FOREGROUND HARDCOPY ****
DSNAME=L00510.X.DATA                                  (LRSP2    )

 PROC RANK DATA=X.P2A OUT=X.P2R GROUPS=4;VAR V1-V12;
DATA X.P2I1;SET X.P2R;BY CODE;
ARRAY N(I) N1-N12; ARRAY F(J) F1-F48; ARRAY V(I) V1-V12;
RETAIN F1-F48 0; KEEP CODE GR N1-N12;
DO I=1 TO 12; J=12*V+I; F=F+1;END;
IF LAST.CODE THEN DO;
 DO GR=0 TO 3;DO I=1 TO 12;J=12*GR+I;N=F;END;OUTPUT;END;
 DO OVER F;F=0;END;
END;
DATA X.P2IN1;SET X.P2I1  END=EOF ;ARRAY N(I) N1-N12; ARRAY H(I) H1-H12;
 RETAIN H1-H12 0;
 DO I=1 TO 12;IF N>0 THEN DO;H=H+N*LOG(N);END;END;
 IF EOF THEN OUTPUT; KEEP H1-H12;
PROC PRINT DATA=X.P2I1 (OBS=3);
PROC PRINT DATA=X.P2IN1;
```

SAS Program to Discretize Filter Outputs.

```
**** TSO FOREGROUND HARDCOPY ****
DSNAME=Z00D16.A.DATA                                    (TSTLN   )

DATA A;SET X.P2;KEEP N;
DATA B;SET X.P2;KEEP V1 V5 V12;
PROC MATRIX;FETCH NS DATA=A;DO K=1 TO 5;
 N=NS(K,); FETCH X N DATA=B;E=X(+,)#/N;X=X-J(N,1,1)*E;
 S=X'*X;S=S#/(N-1);S=INV(S);XA=X*S;X=(X#XA)(,+);FREE XA;
 C=PROBCHI(X,3);C=INT(C#10)+1; F=J(1,10,0);EF=J(1,10,N#/10);
 DO L=1 TO N;F(,C(L,))=F(,C(L,))+1;END;C=(F-EF)*((F#/EF-1)');P=
  PROBCHI(C,9);
 F=(F'//C//P)';PRINT F;END;
PROC MATRIX;FETCH NS DATA=A;DO K=1 TO 5;
 N=NS(K,); FETCH X N DATA=B;X=LOG(X);E=X(+,)#/N;X=X-J(N,1,1)*E;
 S=X'*X;S=S#/(N-1);S=INV(S);XA=X*S;X=(X#XA)(,+);FREE XA;
 C=PROBCHI(X,3);C=INT(C#10)+1; F=J(1,10,0);EF=J(1,10,N#/10);
 DO L=1 TO N;F(,C(L,))=F(,C(L,))+1;END;C=(F-EF)*((F#/EF-1)');P=
  PROBCHI(C,9);
 F=(F'//C//F)';PRINT F;END;
```

SAS Program for Tests of Normality and Lognormality.